Repetition: For loop

- So far we've been dealing with loops that run based on a condition. i.e. the conditional loop

- So what if we wanted to repeat code based on a sequence?
  - That is, we know exactly how many times we want to repeat a chunk of code. (like a top 10 list!)

Definition of a counted loop

- A loop that repeats a predetermined, fixed number of times.
  - This loop is based on an ordered list of things

An example:

```
for c in range(4):
    print (c)
    print (“tag”)
```

- Where *c* is the sentry variable
- The sequence that the loop moves through is defined by *range( )*
  - In this case the loop will repeat from 0 up to (but not including) 4

```
for c in range(4):
    print (c)
    print (“tag”)
```

OUTPUT:

```
0
tag
1
tag
2
tag
3
tag
```

Another Example

```
for c in range(5, 10):
    print (c)
```

- The sequence that the loop moves through is defined by *range( )*
  - In this case the loop will repeat the loop from 5 up to (but not including) 10

```
for c in range(5, 10):
    print (c)
```

OUTPUT:

```
5
6
7
8
9
```

Manipulate the previous *for loop* to do the following:

What if I wanted to print 2 to 5 ?

What if I wanted to print “Hi there” 10 times?

Print 2 to 5:

```
for c in range(2, 6):
    print (c)
```

Print “Hi there” 10 times:

```
for c in range(10):
    print ("Hi There")
```

The decreasing *for loop*

- Definition of a decreasing counted loop
  - same as increasing **for loop** except we count down

  For Example:

```
for c in range(10,0,-1):
    print (c)
```

  - In this case the loop will repeat the loop from the range of 10 down to (but not including) 0

```
for c in range(10,0,-1):
    print (c)
```

OUTPUT:

```
10
9
8
7
6
5
4
3
2
1
```

For loops (jumping)

- A jumping *for* loop is where the sentry value increases by a value greater than 1 for every loop
- i.e. every time the loop executes, the sentry value will increase or decrease by a value other than 1

Trace the following:

```
for a in range (20, 5, -5):
    print (a)

for x in range (5,10, 2):
    print (x)

for c in range(0, 100, 10):
    print (c)
```

Try this!

- Write a program for the following problem:

Print the even numbers from one million down to zero.

Solution

```
for c in range(1000000, -1, -2):
    print (c)
```

Looping Through Strings

We can also use a for loop to access individual characters in a string of text.

```
for c in “ultimate” :  # Prints u, l, t, …, t, e
  print(c)
```

At each iteration of the loop, c takes on the value of the next letter.